Science of Computer Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Formal methods and finite element analysis of hurricane storm surge: A case study in software verification

John Baugh *, Alper Altuntas

Department of Civil, Construction, and Environmental Engineering, North Carolina State University, Raleigh, NC, USA

ARTICLE INFO

Article history:
Received 7 October 2016
Received in revised form 18 August 2017
Accepted 21 August 2017
Available online xxxx

Keywords:
Formal methods
Model checking
Scientific computing
Earth and atmospheric sciences

ABSTRACT

Used to predict the effects of hurricane storm surge, ocean circulation models are essential tools for evacuation planning, vulnerability assessment, and infrastructure design. Implemented as numerical solvers that operate on large-scale datasets, these models determine the geographic extent and severity of coastal floods and other impacts. In this study, we look at an ocean circulation model used in production and an extension made to it that offers substantial performance gains. To explore implementation choices and ensure soundness of the extension, we make use of Alloy, a declarative modeling language with tool support and an automatic form of analysis performed within a bounded scope using a SAT solver. Abstractions for relevant parts of the ocean circulation model are presented, including the physical representation of land and seafloor surfaces as a finite element mesh, and an algorithm operating on it that allows for the propagation of overland flows. The approach allows us to draw useful conclusions about implementation choices and guarantees about the extension, in particular that it is equivalence preserving.

1. Introduction

Coastal flooding from tropical storms is the result of large-scale processes whose simulation is computationally demanding. Ocean circulation models, which attempt to capture the resulting hydrodynamics as accurately and efficiently as possible, employ a variety of mechanisms that, while improving performance, can also lead to complex software implementations. Based on numerical techniques such as finite element methods [41], these models start from a characterization of wind velocities, atmospheric pressure, and land and seafloor surfaces to produce time histories of spatially varying water surface elevations and velocities. Ocean circulation models are designed to perform these large-scale simulations while incorporating both tidal effects and more extreme storm events into the analysis.

As in other areas of science, validation of the models is based on observational comparisons and, as much as anything, review and evaluation by an active community of scientists and engineers. Quantities like wind, wave, and water level are compared with data from the National Data Buoy Center (NDBC), National Ocean Service (NOS) observation stations, high water marks, and other sources [23]. In addition to such studies, since 2008 a regional testbed has been actively comparing the accuracy of ocean circulation models and their relative abilities to hindcast the effects of historical storms [33].

* Corresponding author.
E-mail address: jwb@ncsu.edu (J. Baugh).

http://dx.doi.org/10.1016/j.scico.2017.08.012
0167-6423/

Please cite this article in press as: J. Baugh, A. Altuntas, Formal methods and finite element analysis of hurricane storm surge: A case study in software verification, Sci. Comput. Program. (2017), http://dx.doi.org/10.1016/j.scico.2017.08.012

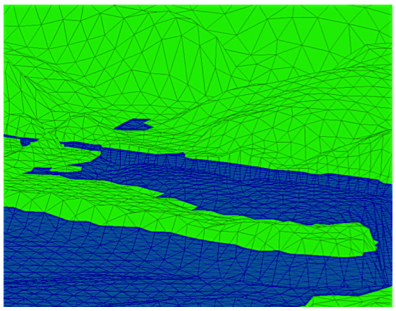

Fig. 1. Finite element mesh.

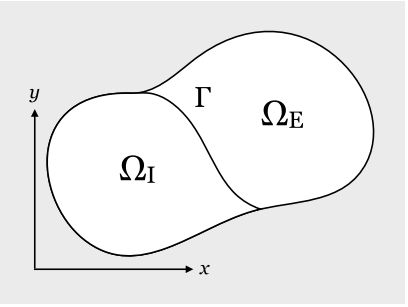

Fig. 2. Domain Ω partitioned at interface Γ.

In this paper, our concern is the correctness of an extension made by our group to ADCIRC [29], an ocean circulation model widely used by the U.S. Army Corps of Engineers and others to simulate hurricane storm surge. ADCIRC itself has been extensively validated against actual flooding conditions, with simulation times of about a thousand or more CPU-hours.

To get a sense of the problem, the finite element *mesh* in Fig. 1 depicts a shoreline extracted from a larger domain that encompasses the western North Atlantic Ocean, the Caribbean Sea, and the Gulf of Mexico. The land and seafloor surfaces are represented as a collection of contiguous, non-overlapping triangles, or *elements*, that meet along their edges and at their vertices, or *nodes*. For this problem, 620 089 nodes and 1 224 714 elements appear in the full mesh. Forced with winds and tides, the physics of the model are realized in three primary routines that are executed in succession at each discrete time step. The first finds the water surface elevations for nodes in the domain using principles of momentum and continuity. Next, the wet–dry state of nodes is determined from empirical rules so that advancing and receding flood waters can be modeled. Finally, velocities are determined using momentum principles, completing one time step.

Our extension, now included in ADCIRC, is an exact re-analysis technique that enables the assessment of local *subdomain* changes with less computational effort than would be required by a complete resimulation [7]. Fig. 2 shows a domain Ω partitioned at interface Γ into a subdomain Ω_I, representing the interior of a geographic region of interest, and Ω_E. The technique starts with a simulation on Ω that produces water surface elevations, velocities, and wet–dry states that are used as *boundary conditions* along interface Γ in subsequent low-cost simulations on Ω_I. We refer to the first simulation on Ω as a *full run*, and the latter one on Ω_I as a *subdomain run*. A correctness condition requires that boundary conditions be imposed in such a way that full and subdomain runs produce equivalent results in region Ω_I.

1.1. Formal methods and verification

The tools and techniques most often associated with scientific computing are those of numerical analysis and, for large-scale problems, structured parallelism to improve performance. Beyond those conventional tools, we also see a role for formal methods and present one such application here using the Alloy language and analyzer [25].

Alloy combines first-order logic with relational calculus and associated quantifiers and operators, along with transitive closure. It offers rich data modeling features based on class-like structures and an automatic form of analysis that is performed within a bounded scope using a SAT solver. For *simulation*, the analyzer can be directed to look for instances satisfying a property of interest. For *checking*, it looks for an instance violating an assertion: a counterexample. The approach is *scope complete* in the sense that all cases are checked within user-specified bounds. Alloy's logic supports three distinct styles of expression, that of predicate calculus, navigation expressions, and relational calculus. The language used for modeling is also used for specifying properties of interest and assertions.

Using Alloy, we model finite element domains and simulations on them so we can experiment with the type of boundary conditions that might be imposed on subdomain runs, as illustrated in Fig. 3. The upper left side of the figure represents ADCIRC as it might be employed without our extension: a Fortran implementation and, beneath it, a model of a simulation on Ω, a full domain. The upper right side of the figure represents our extension: a *possible* implementation and, beneath it, a model of a simulation on Ω_I, a subdomain with boundary conditions imposed on Γ. At the bottom of the figure is a comparison between full and subdomain runs in Alloy, where the assertion *SameFinalStates* is checked to see whether they produce equivalent results in region Ω_I. Through an iterative process, we seek an approach for boundary conditions that satisfies the assertion, and then we make use of the insights gained in an *actual* implementation of our extension, which we refer to as *subdomain modeling*.

We observe that other approaches to verification might also be considered. For instance, a mathematical statement of the equations of motion for bodies of water could be adopted as a specification. Because ocean circulation models are defined by a set of hyperbolic partial differential equations (PDEs) [41], however, asking whether an extension such as subdomain modeling implements it requires the tools of numerical analysis in a labor-intensive process that has already been undertaken for ADCIRC. Instead, we utilize the existing foundation and solve a simpler problem: show that the results of a subdomain run are equivalent to a full run in ADCIRC.

Please cite this article in press as: J. Baugh, A. Altuntas, Formal methods and finite element analysis of hurricane storm surge: A case study in software verification, Sci. Comput. Program. (2017), http://dx.doi.org/10.1016/j.scico.2017.08.012